НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
Кафедра автоматизованих систем обробки інформації та управління

До захисту допущено:
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (вл.ім'я, прізвище)
"___"_____2020 р.

Дипломний проєкт
на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інформаційні управляючі системи та технології»
спеціальності 122 «Комп'ютерні науки та інформаційні технології»

на тему: «Інформаційна система підтримки дистанційної освіти»

Виконав : студент *IV* курсу, групи *ІС-з61*

Сіваченко Владислав Олексійович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник *ст. викл. Родіонов Павло Юрійович* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Консультант з графічної документації *доц., к.т.н., доц. Тєлишева Тамара Олексіївна* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____
(підпис)

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет (інститут) *інформатики та обчислювальної техніки*
(повна назва)

Кафедра *автоматизованих систем обробки інформації та управління*
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки та інформаційні технології»

Освітньо-професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (вл.ім’я, прізвище)
“___” _____2020 р.

ЗАВДАННЯ
на дипломний проєкт студенту

Сіваченко Владислав Олексійович
(прізвище, ім’я, по батькові)

1. Тема проєкту « Інформаційна система підтримки дистанційної освіти »

керівник проєкту Родіонов Павло Юрійович ст.викл
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від “*7*”*травня* 2020 р. №1081-с

2. Термін подання студентом проєкту “*01*”*червня* 2020 року
3. Вихідні дані до проєкту
Технічне завдання

4. Зміст пояснювальної записки
1. Загальні положення: основні визначення та терміни, опис предметного середовища, огляд ринку програмних продуктів, постановка задачі
2. Інформаційне забезпечення: вхідні дані, вихідні дані, опис структури бази даних
3. Математичне забезпечення: змістовна та математична постановки задачі, обґрунтування та опис методу розв’язання
4. Програмне та технічне забезпечення: засоби розробки, вимоги до технічного забезпечення, архітектура програмного забезпечення, побудова звітів
5. Технологічний розділ: керівництво користувача, методика випробувань програмного продукту

5. Перелік графічного матеріалу

1. Схема бази даних

2. Схема структурна класів клієнту

3. Схема структурна класів серверу

4. Схема структурна послідовності

5. Схема структурна компонентів

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання «*13*» *квітня 2020 року*

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	*Вивчення рекомендованої літератури*	*16.02.2019*	
2.	*Аналіз існуючих методів розв'язання задачі*	*23.02.2019*	
3.	*Постановка та формалізація задачі*	*01.03.2019*	
4.	*Розробка інформаційного забезпечення*	*11.03.2019*	
5.	*Алгоритмізація задачі*	*17.03.2019*	
6.	*Обґрунтування використовуваних технічних засобів*	*23.03.2019*	
7.	*Розробка програмного забезпечення*	*28.03.2019*	
8.	*Налагодження програми*	*03.04.2019*	
9.	*Виконання графічних документів*	*11.04.2019*	
10.	*Оформлення пояснювальної записки*	*23.04.2019*	
11.	*Подання ДП на попередній захист*	*15.05.2020*	
12.	*Подання ДП на основний захист*	*01.06.2020*	
13.	*Подання ДП рецензенту*	*02.06.2020*	

Студент Сіваченко В. О.

Керівник Родіонов П. Ю.

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	ДП 6121.00.000 ПЗ	Пояснювальна записка	87	
3	А4	ДП 6121.01.000 ТЗ	Технічне завдання	8	
4	А4	ДП 6121.04.000 СБД	Схема бази даних	1	
5	А4	ДП 6121.05.000 ССК	Схема структурна класів програмного	2	
			забеспечення		
6	А4	ДП 6121.06.000 ССП	Схема структурна послідовності	1	
7	А4	ДП 6121.07.000 СК	Схема структурна компонентів	1	
			програмного забеспечення		

Зм.	Арк.	ПІБ	Підп.	Дата	ДП 6121 00.000.00		
Розробн.		Сіваченко В.О.			Відомість дипломного проєкту	Літ.	Лист 1 / Листів 1
Керівн.		Радіонов П.Ю.					
Консульт.						КПІ ім. Ігоря Сікорського Каф. АСОІУ Гр. ІС-з61	
Н/контр.		Тєлишева Т.О.					
В.о.зав.каф.		Павлов О.А.					

Пояснювальна записка
до дипломного проекту

на тему: Інформаційна система підтримки дистанційної освіти

Київ – 2020 року

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проекту складається з шести розділів, містить 15 рисунків, 9 таблиць, 1 додатків, 10 джерел.

Дипломний проект присвячений розробці комплексу задач для підтримки дистанційної освіти.

У розділі інформаційного забезпечення описані можливі вхідні та вихідні дані, опис моделі бази даних та структура масивів інформації.

Розділ математичного забезпечення присвячений постанові задачі та пошуку, обґрунтуванню рішень поставлених задач.

Програмне забезпечення та вимоги до нього розкриті у розділі «Програмне та технічне забезпечення».

У технологічному розділі описані тести та випробування програмного продукту для знаходження недоліків.

СТУДЕНТ, ПЛАНУВАННЯ, ІНФОРМАЦІЙНА СИСТЕМА, ТЕОРІЯ РОЗКЛАДІВ, ПЕРЕРОЗПОДІЛ, ДОСЛІДЖЕННЯ ОПЕРАЦІЙ

ABSTRACT

Structure and scope of work. The explanatory note of the diploma project consists of six sections, contains 15 drawings, 9 tables, 1 applications, 10 sources.

The diploma project is devoted to the development of a set of tasks to support distance education.

The information support section describes possible input and output data, a description of the database model and the structure of information arrays.

The section of mathematical support is devoted to the solution of the problem and search, substantiation of the solutions of the set problems.

The software and its requirements are disclosed in the "Software and hardware" section.

The technological section describes the tests and trials of the software product to find defects.

STUDENT, PLANNING, INFORMATION SYSTEM, THEORY OF SCHEDULES, REDISTRIBUTION, INVESTIGATION OF OPERATIONS

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – набір чітко визначених методів для взаємодії різних компонентів.

Компонент – незалежний блок інтерфейсу, що може бути ізольованим від іншої частини програми та повторно використаним на інших сторінках чи в іншому додатку.

HTTP – це клієнт-серверний протокол, тобто запити відправляються якоїсь однієї стороною - учасником обміну (user-agent) (або проксі замість нього). Найчастіше в якості учасника виступає веб-браузер, але їм може бути хто завгодно, наприклад, робот, який мандрує по Мережі для поповнення та оновлення даних індексації веб-сторінок для пошукових систем.

REST – це фрхітектура, тобто принципи побудови розподілених гіпермедіа систем, того що іншими словами називають World Wide Web, включаючи універсальні способи обробки і передачі станів ресурсів по HTTP.

ВСТУП

Сьогодення диктує нам свої сурові правила. Кожен вищий навчальний заклад намагається закріпитись на мапі України та набрати якомога більше студентів.

Для цього університет має створити всі умови для абітурієнта та студента, аби залишатись на верхній сходинці серед інших представників. Саме для цього створюються різноманітні навчальні програми та лабораторії, унікальні програмні продукти та безпосередньо сайти на базі університетів.

Навчання студентів – це дуже важливий та кропіткий процес. Саме для цього навчальний процес у вищих навчальних закладах має особливі методи: лекції, практичні та лабораторні заняття, для покращення практичних знань, самостійну роботу студентів, практичну підготовку студентів для підвищення кваліфікації, курсове та дипломне проектування для оцінювання знань накопичених студентом. Все це неможливо без практичних занять.

У закладах середньої освіти ситуація інша. Тут немає практичних робіт, а весь матеріал є лише теоритичним. Через це учні після навчання у школі, немають не тільки практичних знань, а нормального рівня знань теоритичного матеріалу, через відсутність нормальної системи оцінювання. Тут треба розуміти, що проблема навіть не стільки у вчителях або учнях, проблема у системі навчального процесу та оцінюванні знань в закладах середньої освіти.

Проблема оцінювання знань у закладах середньої освіти, полягає у тому, що:

- немає оцінювання практичних знань учнів;
- оцінка теоритичних знань дуже субьєктивна;
- недостатньо матеріалу для повного розкриття теми;
- некомпитентність деяких вчителів.

Оцінки, отримані за окремі практичні заняття, навідміну від школ, враховуються при виставленні підсумкової за предмет, та є підставою покращення загального балу.

Дидактична мета практичного заняття – розширення, поглиблення й деталізація наукових знань, отриманих студентами на лекціях та в процесі самостійної роботи і спрямованих на підвищення рівня засвоєння навчального матеріалу, формування умінь і навичок, розвиток наукового мислення та усного мовлення студентів[3].

Огранізація практичних занять є важливою частоною будь якого навчального процесу, та стають основою дидактичного принципу, який існує для зв’язку теорії з практикою, та вирішує більшість проблем:

- систиматизація теоритичних знань учнів;
- набування практичних умінь та знань;
- можливість отримання досвіду в окремій галузі;
- самостійний розвиток.

Тепер поговоримо про сферу відповідальності студента, яка розкривається у десктопному та мобільному застосунках, а саме самостійна робота студента.

Випусники вищих навчальних закладів мають самостійно допрацьовувати та поповнювати свої знання і творчо застосовувати їх у своїй практичній діяльності. У цьому контексті важливе місце відводиться вдосконаленню самостійної роботи студентів, її організації і плануванню яка формує фахівця завдяки індивідуального підходу, коли найбільш повно розкриваються здібності суб’єкта навчання, реалізується його творчий потенціал.

Самостійна робота – це основний засіб засвоєння навчального матеріалу.

Зміст самостійної роботи визначається змістом навчання з

орієнтацією на освітньо-кваліфікаційну характеристику. Зміст самостійної роботи до конкретної навчальної дисципліни визначається робочою навчальною програмою та методичними рекомендаціями викладача.

Самостійні роботи у закладах середньої освіти мають зовсім інше значення та зазвичай є просто перевіркою матеріалу засвоєного учнем. Але це у корні не вірний підхід до засвоєння матерілу. Самостійна робота повинная проводитися для засвоєння, а не для перевірки. У «школах» найчастіше, самостійну роботу порівнюють з тестом знань та цим руйнують концепцію самостійного навчання учнів. Далі ми порівняємо цю концепцію з концепцією самостійних робот у вищіх навчальних закладах та побачимо чим вони відрізняются.

Метою самостійної роботи студента є втілення в життя принципу індивідуального підходу до навчання; крім того, регламентована навальним планом і здійснювана під керівництвом викладача самостійна робота студента може стати найефективнішим методичним засобом реалізації цього принципу. Тобто самостійна робота студента у широкому розумінні – це робота, що стосується оволодіння науковими знаннями та практичними уміннями та навичками, активна розумова діяльність в усіх формах навчально-виховного процесу.

Приклади самостійної роботи студента:

- підготовка до лекцій та практикумів;
- підготовка до лабораторних;
- виконання індивідуальних завдань з дисципліни;
- виконання тестів для самоконтролю.

Продукт має бути у вільному доступі для студента, аби він мав змогу працювати над завданням з будь-якої точки країни. Також потрібно виділити інтерфейс.

Він має бути доступним, та якомога зручнішим. Це означає, якщо Ви відкриваєте сайт університету з телефону або комп’ютера, Ви не відчуваєте

різниці, ніби, все знаходиться на своїх місцях.

Проаналізувавши більшість систем дистанційного навчання, було сформовано основні проблеми цих системи:

- платформи на якій базується система;
- розміщення та збереження матеріалу;
- слідкування за навчальним процесом;
- оновлення матеріалу;
- можливість перевірки робіт;
- можливість розширення функціоналу системи.

Вирішення цих проблем та розробка системи дистанційного навчання, з їх урахуванням і є дипломної роботою[3].

Цілі та задачі розробки

Метою розробки є покращення навчального процесу у закладах середньої освіти.

Задачі :

- створення бази даних;
- розробка WebApi сервісу;
- розробка Desktop застосунку;
- з’єднання застосунків з сервісом;
- автоматизація оновлення та синхронізації.

Для вирішення поставлених задач, була проаналізована робота такого сервісу як Moodle. На основі отриманих даних стало зрозуміло, що вирішення деяких проблем, наприклад контроль успішності, може відійти на другий план перед проблемою надання матеріалу та можливості перевірки робіт.

Для збереження робіт та задач, було обрано незалежне онлайн джерело, що дозволяє навчальному закладу, не мати свого сховища взагалі, та працювати безпосередньо з хмарним сховищем, наприкла «Google Drive».

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Предметною областю є робота закладу середньої або початкової освіти, зокрема автоматизація навчального процесу.

Основною метою автоматизації навчального процесу є зменшення навантаження на викладачів, прискорення роботи навчального закладу та якості надання матеріалу.

Можливість проводити контрольні та самостійні роботи онлайн. Надання домашніх робіт. Відказ від потрібності зошитів для домашніх робіт або конспектів. Всі матеріали та роботи будуть зберігатися у базі даних та надаватися викладачам та учням за потребою. Школи більше не потребуватимуть підручники для надання матеріалу.

Ведення журналу та ручне заповнення, можливість впливу викладачів на успішність учнів через особисті мотиви буде зведена до мінімуму через автоматизацію більшості компонентів навчального процесу.

Буде можливість контролю успішності учнів їх батьками завдяки онлайн сервісу та повідомлення про всі новини навчального закладу, без необхідності батьківських зборів.

Можливість контролю відвідувань занять учнями та їх поведінки. Надання інформації про заміни та відсутність деяких уроків через особливі обставини.

1.1.1 Опис процесу діяльності

Планується автоматизація таких процесів як:

- надання та приймання домашнього завдання;
- проведення самостійних та контрольних робіт;
- підрахунок поточних та підсумкових балів учнів;
- ведення статистики успішності та поведінки учнів;

- надання матеріалу з уроків що біли проведені або ще будуть;
- надання матеріалу за конкретною темою;
- проведення батьківських зборів та контроль успішності дітей.

1.1.2 Опис функціональної моделі

Для реалізації функціоналу системи, було створено декілька типів користувачів. Акторів системи та їх функції можно переглянути у таблиці 1.1

Таблиця 1.1 – Функціональна модель

Актор	Функція
Вчитель	Можливість створення та перевірки домашніх, самостійних та контрольних робіт. Контроль успішності та поведінки учнів. Виставлення поточних балів. Надання матеріалу для учнів. Перегляд новин закладу.
Учень	Можливість проходження самостійних та контрольних робіт. Надання домашніх робіт онлайн. Перегляд погрібного матеріалу. Контроль успішності та поведінки. Перегляд новин закладу.
Адміністратор	Всі права вчителів та учнів. Повних контроль учбового процесу. Можливість вносити зміни до онлайн журналу. Створення аккаунтів та надання їм потрібних прав.

1.2 Огляд наявних аналогів

У кожного навчального закладу є свої сайти або базові варіанти застосунків але вони не мають потрібного функціоналу та не можуть використовуватись для повної автоматизації навчального процесу.

Більшість подібних аналогів існують для надання розкладу та мінімальної інформації про навчальний заклад.

1.3 Постановка задачі

1.3.1 Призначення розробки

Призначення цієї розробки є автоматизації навчального процесу, зменшення навантаження на викладачів, прискорення роботи навчального закладу та якості надання матеріалу.

1.3.2 Цілі та задачі розробки

Метою розробки є покращення навчального процесу у закладах середньої освіти.

Задачі :

- створення бази даних;
- розробка WebApi сервісу;
- розробка Desktop застосунку;
- з’єднання застосунків з сервісом;
- автоматизація оновлення та синхронізації.

Висновок до розділу

Таке програмне забезпечення повинно в декілька разів пришвидшити офіційну частину навчального процесу та покращити його якість, шляхом зменшення ризиків помилки та суб’єктивізму.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Вхідні дані

Вхідні дані представлені у таблиці 2.1.
Таблиця 2.1 – Вхідні дані

Дані	Опис
Дані про користувача	Всі відомі дані про користувача (ім’я, прізвище, email, день народження, логін та пароль)
Дані про учнів	Посилання на клас в якому знаходиться учень та посилання на користувача
Дані про вчителів	Посилання на предмет який викладає вчитель та посилання на користувача
Дані про класи	Містить назву та номер класу
Дані про предмети	Містить назву предмету
Дані про заняття	Містить назву та матеріал заняття, а також номер класу для якого воно проводиться
Дані про домашню роботу	Містить назву роботи, номер класу для якого вона розроблена та посилання на зовнішнє джерело з самою роботою

2.2 Вихідні дані

Вихідні дані представлені в таблиці 2.2.
Таблиця 2.2 – Вихідні дані

Дані	Опис
Дані журналу оцінок	Середній бал учнів на теперішній стан, оновлюються вчителем
Дані виконаних домашніх робіт	Містить посилання на роботу, яку виконано, посилання на учня який виконав та на зовнішнє джерело

2.3 Опис структури бази даних

Структура бази даних зображена у вигляді діаграми ERD.

На діаграмі можна побачити схему таблиць БД та їх зв’зки:

- таблиця User, на яку посилаються таблиці Student та Teacher,тому що ці таблиці мають одинакові властивості та вважаються користувачами системи.
- таблиця Class об’єднує студентів за номером та літерою які позначаються клас.
- таблиця Subject зберігає в собі назву та ID предмету.
- таблиця Lesson зберігає в собі посилання на предмет(Subject), назву предмету, матеріали до предмету у вигляді тексту.
- таблиця Homework зберігає в собі назву домашньої робити, посилання на предмет за яким було видано завдання та посилання на зовнішнє джерело з файлом домашньої роботи.
- таблиця Homeworld_Done зберігає в собі посилання на студента, котрий виконав завдання, посилання на домашнє завдання та посилання на зовнішнє джерело з виконаним домашнім завданням у вигляді файлу.

– таблиця Journal зберігає в собі посилання на студента, який отримав оцінку за завдання, посилання на предмет, за яким було надано оцінку, та безпосередньо сама оцінка у вигляді цілого числа.

Рисунок 2.1 – ERD діаграма структури БД

Висновок до розділу

Вхідні та вихідні дані для даного програмного продукту є більш абстрактні та являють собою якість зв’язку та швидкість роботи сервісу.

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Змістовна постановка задачі

Постанова задачі: розробити інформаційну систему для підтримки дистанційної освіти школярів. Додаток має бути інтуїтивно зрозумілим як для учнів, так і для вчителів. Мати весь необхідний функціонал, описаний вище, та мати можливість адаптації під умови учбового закладу.

3.2 Математична постановка задачі

Необхідно знайти швидкий та гнучкий метод роботи з базою даних. Забезпечити можливість не роботи із різними платформами та різними клієнтами. Створити універсальний сервер з можливістю розширення функціоналу та зміни існуючих можливостей.

3.3 Обґрунтування методу розв’язання

Методом розв’язання даної задачі було обрано метод взаємодії клієнта з сервером та обміну даних з БД через застосування HTTP/REST запитів. Метод було обрано через його швидкість та стабільність обміну команд даних із сервером, серед недоліків можна виділити тільки сумнівний захист даних.

3.4 Опис методів розв’язання

HTTP протокол описує взаємодію між двома комп'ютерами (клієнтом і сервером), побудоване на базі повідомлень, званих запит (Request) і відповідь (Response). Повідомлення складається з частин:

- стартова рядок;
- заголовки;
- тіло.

При цьому обов'язковою є тільки стартова рядок.

Стартові рядки для запиту і відповіді мають різний формат - нам цікава тільки стартова рядок запиту, яка виглядає так: де METHOD - цеяк раз метод HTTP-запиту, URI - ідентифікатор ресурсу, VERSION - версія протоколу[10].

Заголовки - це набір пар ім'я-значення, розділених двокрапкою. У заголовках передається різна службова інформація: кодування повідомлення, назву та версію браузера, адреса, з якого прийшов клієнт (Referrer) і так далі. Тіло повідомлення - це, власне, передані дані. У відповіді переданими даними, як правило, є html-сторінка, яку запросив браузер, а в запиті, наприклад, в тексті листа передається вміст файлів, що завантажуються на сервер. Але як правило, тіло повідомлення в запиті взагалі відсутня.

Ресурси і методи. Повернемося до стартової рядку запиту і згадаємо, що в ній присутній такий параметр, як URI. Це розшифровується, як Uniform Resource Identifier - однаковий ідентифікатор ресурсу. Ресурс - це, як правило, файл на сервері (приклад URI в даному випадку '/styles.css'), але взагалі ресурсом може бути і якийсь абстрактний об'єкт ('/ blogs / webdev /' - вказує на блок «Веб- розробка », а не на конкретний файл). HTTP-запити можуть вказувати на дію яку ми саме плануємо зробити. Напочатку планувалося лише отримання даних, але даному етапі розвитку протоколу, можно додавати, редагувати та видаляти дані.

REST (передача стану уявлення) - стиль взаємодії між компонентам системи. REST дозволяє будувати систему більш продуктивною та простою. Він також пришвидшує розробку та робить систему більш адаптивною для змін та оновлень.

Властивості REST-системи:

- незалежність оновлення єлементів системи;
- можливість;
- продуктивність системи;
- масштабованість компонентів.

Висновок до розділу

Розроблена система може адаптуватися під будь які умови застосування, завдяки універсальному серверу, клієнт може бути написаний на будь якій мові та з використання будь яких технологій розробки під будь яку платформу.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

Програма розроблена в середовищі Microsoft Visual Studio 2019, на мові C#. Клієнтська частина розроблялась за допомогою технології WPF (Windows Presentation Foundation), вона узяла за основу векторну графіку та технологію DirectX, що дозволяє їй бути швидшою за існуючи аналоги та мати сучасну графіку. У WPF використовується мова графічного інтерфейса XAML, яка базується на XML та є аналогом HTML для десктопних застосунків. XAML має графіний інтерфес, що дозволяє контролювати процесс створення вікон та сторінок додатку у режимі реального часу. WPF реалізує систему binding data, що дозволяє прив’язати дані до графічного інтерфейсу саме в графічному інтерфейсі. У такому випадку WPF та XAML бере на себе відстеження зміни даних та внесення їх до моделі, що дозволяє пришвидшити як саму розробку, так і роботу додатку в цілому.

Для роботи з сервером на «клієнтах» існують контролери, які можуть відправляти та отримувати дані з сервера. Передача даних відбувається у Json форматі, та завдяки HTTP протоколу. Контролер, має свою структуру даних, завдяки чому знає то якого типу треба привести Json після отримання.

Серверна частина розроблена за допомогою технологій WebApi та HTTP/REST запитів. WebApi є частино. ASP.NET технології та існує для відокремлення логіки роботи з даними та графіного інтерфейсу. Вона дозволяє будувати систему клієнт – серверної архітектури. По суті WebApi сервер є набором контролерів та моделей для работи за даними та обміном інформацією з клієнтом та базаю даних. Також саме на сервері відбувається обробка та аналіз даних отриманих з бази або клієнта.

Для роботи з базою даних використовувався Entity Framework, як частина технології ADO.NET. Ця совокупність технологій дозволяє мати

об’єктно – орієнтований доступ до даних Тому розробнику не потрібно самостійно робити запити до бази та використовувати мову SQL. Entity Framework створює модель бази даних, та дозволяє роботати з нею як зі звичайними екземлярами класів. А завдяки LINQ, можно досягти деталізації запитів не гірше ніж у T-SQL.

У роботі з Entity Framework використовувався принцип «Database First», що дозволило створити базу окремо від системи и потім з’єднити їх за допомогою технології створення моделі бази даних.

Загальних список мов та технологій які були використані:

- мова програмування C#;
- технологія WPF;
- мова графічного інтерфейсу XAML;
- передача запитів за допомогою HTTP протоколів та Json формату;
- серверна технологія WebApi на базі технології ASP.NET;
- технологія для роботи з базою даних Entity Framework;
- LINQ для роботи з моделями та запитами до них.

4.2 Вимоги до технічного забезпечення

4.2.1 Загальні вимоги

«Клієнтська» части розроблена на технології WPF тому, користувач повинен використовувати комп’ютер на базі операційної системи Windows 7, 8, 10 та з наступними конфігураціями:

- процесор Intel Pentium 2 ГГц і вище;
- мінімальний об’єм оперативної пам’яті 512 Мб;
- об’єм вільної пам’яті на пристрої від 50 мб;
- для деяких функцій наявність інтернет зв'язку.

4.2.2 Опис локальної обчислювальної мережі

База даних та WebApi сервіс знаходяться на зовнішньому хостінгу, а тому для роботи мережі необхідно лише мати доступ до інтернету, клієнт встановить зв’язок із сервісом, той в свою чергу буде працювати з базою даних у від’єднаному режимі.

4.3 Архітектура програмного забезпечення

4.3.1 Діаграма класів

Структурні схеми класів клієнтської та серверної частин, наведена в графічному матеріалі.

У схемі наведені всі класи клієнтської та серверної частин інформаційної системи. Наведені всі методи та поля що містяться в класах.

4.3.2 Діаграма послідовності

На рисунку 4.1 наведена діаграма послідовності для процесу авторизації.

Рисунок 4.1 – Діаграма послідовності для процесу авторизації

Саме ця діаграма є найбільш показовою, бо саме з цього процесу починається робота з системою, а також саме у цьому процесі задіяні усі елементи системи[5].

4.3.3 Діаграма компонентів

На рисунку 4.2 наведена діаграма компонентів системи та їх взаємодія.

Рисунок 4.2 – Діаграма компонентів

4.3.4 Специфікація функцій

Функції наведено з серверної частини системи, бо клієнтська частина може бути будь якою, та не має якихось обмежень в розробці.

Таблиця 4.1 – Специфікація функцій

Назва класу	Назва методу	Дія
MainLogic	AddHomework	Додавання домашньої роботи для виконання
MainLogic	AddMaterial	Додавання матеріалу уроку

Продовження таблиці 4.1.

Назва класу	Назва методу	Дія
MainLogic	DeleteHomework	Видалення домашньої работи
MainLogic	GetClass	Повертає клас окремого учня
MainLogic	GetHomework	Повертає список існуючих домашніх робіт
MainLogic	GetHomework_Done	Повертає список існуючих виконаних робіт
MainLogic	GetJournal	Повертає список оцінок учнів за усіма предметами
MainLogic	GetMaterials	Повертає список уроків які були проведені та матеріали до них
MainLogic	GetSubjects	Повертає список предметів
MainLogic	MainLogic	Конструктор класу, виконує ініціалізацію бази даних
MainLogic	PutHomework	Зміна вже існуючої домашньої роботи
MainLogic	PutHomework_Done	Додавання виконаної роботи учня
MainLogic	PutJournal	Зміна оцінок в журналі

Продовження таблиці 4.1.

Назва класу	Назва методу	Дія
MainLogic	PutMaterial	Зміна матеріалу існуючого заняття
AuthorizationLogic	AuthorizationLogic	Конструктор класу, виконує ініціалізацію бази даних
AuthorizationLogic	GetAuthorization	Виконує спробу авторизації у системі
AuthorizationLogic	GetRestoreAccess	Виконує відновлення даних користувача, завдяки існуючому email
AuthorizationLogic	SendEmail	Відправляє запит на email для відновлення паролю та логіну

В таблиці не наведені методи контролерів, бо вони дублюють методи основних логічних класів системи і по своїй суті є лише методами відправки даних на клієнти.

Висновок до розділу

В даному розділі було розглянуто засоби та мови програмування, за допомогою яких було реалізовано програмний продукт. Приведені структурні схеми класів, послідовності та розгортання. В таблиці приведено опис компонентів діаграми класів.

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

5.1 Керівництво користувача

Щоб показати процес взаємодії додатку з користувачем наведемо приклади виконання додатку. На рисунках 5.1-5.12 приведені скріншоти роботи з програмою

Після запуску застосунку з’являється вікно авторизації у системі.

Рисунок 5.1 – Вікно авторизації

Поля містять валідацію та повідомляють про помилку користувача. Кнопка «Forgot Password?» переводить користувача на вікно відновлення паролю

Рисунок 5.2 – Вікно відновлення паролю

У полі Email потрібно вказати ваш email, який був при створенні вашого облікового запису. При натисканні кнопки «Enter» система спробує знайти ваш email у базі та відновити логін та пароль. При натисканні кнопки «Cansel» система поверне вас на вікно авторизації.

Після авторизації ви потрапляєте на головне меню, де можете бачити своє ім’я та основні пункти меню.

Рисунок 5.3 – Головне меню

Переходячи по пунктам меню мі будемо бачити динамічний контент вікна. У учнів та вчителів він буде відрізнятися через різні потреби від системи.

Рисунок 5.4 – Контент «Матеріали уроків» (учень)

У додатковому меню ми бачимо список уроків та вибравши один з них з’являється матеріал по ньому.

Рисунок 5.5 – Контент «Матеріали уроків» (вчитель)

У вчителя також є можливість додавання та видалення уроків та зміна матеріалів існуючого уроку. При натисканні кнопки «Удалить урок» буде видалено вибраний урок та його матеріал. Кнопка «Сохранить» зберігає зміни у матеріалі уроку. При натисканні «Добавить новый урок» відкриється нове вікно з можливістю додати новий урок, вибрати предмет цього уроку, його назву, матеріал та клас до якого він відноситься.

Рисунок 5.6 – Вікно додавання уроку

Поле «Предмет» являє собою список існуючих предметів. Кнопка «Додати» завершує процес створення уроку та повертає. Вас на попереднє вікно.

При виборі пункту «Домашние работы» ми бачимо список існуючих робіт для нашого класу, якщо ви учень, та при виборі конкретної роботи маємо можливість відкрити її за допомогою кнопки «Открыть» та відправити вчителю свою готову роботу натиснувши «Сдать» та заповнивши поля у вікні що відкрилось.

Рисунок 5.7 – Контент «Домашні роботи» (учень)

Рисунок 5.8 – Вікно здачі домашньої роботи

Якщо користувач є вчителем, то контент буде відрізнятися. Вчитель бачить усі роботи, всіх учнів, має можливість видаляти роботи, додавати нові та перевіряти роботи учнів.

Рисунок 5.9 – Контент «Домашні роботи» (вчитель)

Рисунок 5.10 – Вікно додавання домашньої роботи

При виборі роботи та натисканні кнопки «Проверить» з’являється список виконаних робіт учнів та вчитель має можливість переглянути їх та оцінити.

Рисунок 5.11 – Список учнів, що виконали роботу

При переході на вкладку «Журнал» користувач може бачити поточний бал учнів за предметами, учні можуть дивитися лише бали свого класу, вчитель має можливість перегляду та зміни балів будь якого з учнів.

Рисунок 5.12 – Контент «Журнал»

5.2 Випробування програмного продукту

5.2.1 Мета випробувань

Мета випробувань – це перевірка функцій комплексу задач підтримки дистанційної освіти до вимог технічного завдання[1].

5.2.2 Загальні положення

Випробування проводилися на основі документів:

– ГОСТ 34.603–92. Інформаційна технологія. Види випробувань автоматизованих систем;
– ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5.2.3 Результати випробувань

Були протестовані функціональні можливості системи. У наступних таблицях приведені випробування деяких функціональних можливостей:

Таблиця 5.1 – Процес авторизації

Мета тесту	Перевірка функцій «Авторизації»
Початковий стан моделі	Відкрите вікно входу в систему
Вхідні дані:	Логін та пароль
Схема проведення тесту:	Ввести вхідні дані. Натиснути кнопку «Login in»
Очікуваний результат:	Успішне проходження авторизації та перехід на головне вікно системи
Стан моделі після проведення випробувань:	Відкрите головне меню системи та показано ім’я користувача

Таблиця 5.2 – Перевірка відновлення даних

Мета тесту	Перевірка функцій «Відновлення даних»
Початковий стан моделі	Відкрите вікно відновлення
Вхідні дані:	Email
Схема проведення тесту:	Ввести вхідні дані. Натиснути кнопку «Enter»
Очікуваний результат:	Успішне відновлення та отримання даних на вказану пошту
Стан моделі після проведення випробувань:	Повернення на вікно авторизації, отримання на пошту листа з вашими даними

Таблиця 5.3 – Перевірка додавання матеріалу

Мета тесту	Перевірка функцій «Додавання уроку»
Початковий стан моделі	Відкрите вікно створення уроку
Вхідні дані:	Назва уроку, матеріал, предмет та клас
Схема проведення тесту:	Ввести вхідні дані. Натиснути кнопку «Добавить»
Очікуваний результат:	Повернення на вікно зі списком уроків та вдале додавання нового уроку до бази
Стан моделі після проведення випробувань:	Повернення на вікно матеріалів, у списку вже відображається новий урок з матеріалом

Таблиця 5.4 – Перевірка додавання домашньої роботи

Мета тесту	Перевірка функцій «Додавання домашньої роботи»
Початковий стан моделі	Відкрите вікно додавання роботи
Вхідні дані:	Назва роботи, посилання на роботу, предмет та клас

Продовження таблиці 5.4.

Мета тесту	Перевірка функцій «Додавання домашньої роботи»
Схема проведення тесту:	Ввести вхідні дані. Натиснути кнопку «Добавить»
Очікуваний результат:	Повернення на вікно зі списком робіт та вдале додавання нової роботи до бази
Стан моделі після проведення випробувань:	Повернення на вікно домашніх робіт, у списку вже відображається нова робота з посиланням

Таблиця 5.5 – Перевірка видалення уроку

Мета тесту	Перевірка функцій «Видалення уроку»
Початковий стан моделі	Відкрите вікно зі списком уроків
Вхідні дані:	Вибраний урок
Схема проведення тесту:	Вибрати урок зі списку. Натиснути кнопку «Удалить урок»
Очікуваний результат:	Видалення уроку зі списку та з бази
Стан моделі після проведення випробувань:	Відкрите вікно зі списком уроків, вибраний урок видалено

Висновок до розділу

В даному розділі проведено тестові варіанти з початковими умовами та результатами проходження тестів та на яких підставах вони проводилися.

ЗАГАЛЬНІ ВИСНОВКИ

Головна задача програмного забезпечення в декілька разів пришвидшити офіційну частину навчального процесу та покращити його якість, шляхом зменшення ризиків помилки та суб’єктивізму.

Під час розробки стає зрозумілим, що вхідні та вихідні дані є більш абстрактні та являють собою якість зв’язку та швидкість роботи сервісу, що яскраво відображає створене програмне забезпечення.

Також можна виділити таку сильну сторону системи, як адаптація під будь які умови застосування, завдяки універсальному серверу, клієнт може бути написаний на будь якій мові та з використання будь яких технологій розробки під будь яку платформу, що дуже спрощує життя будь-якого користувача системи.

Під час написання дипломної роботи та розробки продукту було розглянуто декілька технологій та мов програмування. Приведені структурні схеми класів серверу, класів клієтської частини, послідовності та компонентів. В таблиці приведено опис класів системи, які у повному обсязі описують механізми, за якими працює програмне забезпечення.

Останнім у списку, але не останнім за значенням було описано керівництво користувача та проведено тестові варіанти з початковими умовами, вхідними даними та результатами проходження тестів. Усі тести були пройдені із позитивним результатом. Багів або будь-яких інших проблем не було зафіксовано[2].

Під час розробки було реалізовані поставлені задачі.

Задачі :

- створення базиданих;
- розробка WebApi сервісу;
- розробка Desktopзастосунку;
- з’єднання застосунків зсервісом;

ПЕРЕЛІК ПОСИЛАНЬ

1. *Блэк Рекс. Ключевые процессы тестирования. Планирование, подготовка, проведение, совершенствование. / Блэк Р. – М. : Лори, 2011. – 544с.*

2. *ДСТУ 3582:2013 Бібліографічний опис. Скорочення слів і словосполучень українською мовою. Загальні вимоги та правила. Київ: Мінекономрозвитку України, 2014. – 15с.*

3. *Планування і організація навчально-виховного процесу у вищій школі : навч.-метод. посіб. для магістрантів спец. 8.18010021 «Педагогіка вищої школи» / Л. Г. Кайдалова, Н. Б. Щокіна. – Х. : НФаУ, 2014. – 108 с.*

4. *Шилдт Г. C# 4.0: полное руководство.: Пер. с англ. – М.: ООО "И.Д. Вильямс", 2011. – 1056 с.*

5. *Creatly - сервіс для створення діаграм онлайн [Електронний ресурс] – Режим доступу до ресурсу:* https://app.creately.com/diagram/L6aCFjMy5UP/view.

6. *Об'єктно-орієнтоване програмування(C#) [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/concepts/object-oriented-programming.*

7. *Діаграма станів [Електронний ресурс] – Режим доступу до ресурсу: http://flash.retejo.info/cxefpaqxo/uml/diagrama-staniv.*

8. *Введення в Web API [Електронний ресурс] – Режим доступу до ресурсу:* https://metanit.com/sharp/mvc/12.1.php.

9. *Керівництво по WPF [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/sharp/wpf/.*

10. *Типи HTTP-запитів і філософія REST [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/50147/.*

Додаток А

Тексти програмного коду

Інформаційна система підтримки дистанційної освіти

(Найменування програми (документа))

DVD-R

(Вид носія даних)

48 арк, 244 Кб

(Обсяг програми (документа) , арк.,) Кб)

Київ – 2020 року

MainController.cs

```
public class MainController
    {
        SettingServer settingServer;
        public List<LessonsView> lessonsViews;
        public List<HomeworkView> homeworkViews;
        public List<JournalView> journalViews;
        public List<HomeworkDoneTeacherView>
homeworkDoneTeacherViews;

        public MainController()
        {
            settingServer = new SettingServer();
        }

        /// <summary>
        /// Метод для авторизации форма AuthorizationWindow
        /// </summary>
        public User GetAuthorization(string name, string password)
        {

            try
            {
                User userView;
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetAuthorization?userLogin={name}&userPa
ssword={password}").Result;
                response.EnsureSuccessStatusCode();
```

```
                userView = response.Content.ReadAsAsync<User>().Result;
                return userView;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
            }
            return null;
        }

        /// <summary>
        /// Метод для восстонавления логина и пароля через email если он
есть в БД форма AuthorizationWindow
        /// </summary>
        public string GetRestoreAccess(string email)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetRestoreAccess?email={email}").Result;
                response.EnsureSuccessStatusCode();
                return response.Content.ReadAsAsync<string>().Result;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.InnerException.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
```

```
                //Environment.Exit(-68071);

            }
            return "";
        }

        #region Material

        public int GetClass(int Id)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetClass?Id={Id}").Result;
                response.EnsureSuccessStatusCode();
                return response.Content.ReadAsAsync<int>().Result;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                return 0;
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void GetMaterial()
        {
```

```
            try
            {
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetMaterials").Result;
                response.EnsureSuccessStatusCode();
                lessonsViews =
response.Content.ReadAsAsync<List<LessonsView>>().Result;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void PutMaterial(LessonsView view)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.PutAsJsonAsync($"api/PutMaterials", view).Result;
                response.EnsureSuccessStatusCode();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
```

```
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void PostMaterial(LessonsView view)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.PostAsJsonAsync($"api/PostMaterial", view).Result;
                response.EnsureSuccessStatusCode();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void DeleteMaterial(int Id)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.DeleteAsync($"api/DeleteMaterials?Id={Id}").Result;
                response.EnsureSuccessStatusCode();
```

```
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public List<string> GetSubjects()
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetSubjects").Result;
                response.EnsureSuccessStatusCode();
                return response.Content.ReadAsAsync<List<string>>().Result;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                return null;
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }
```

```
        #endregion

        #region Homework

        public void GetHomework()
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetHomework").Result;
                response.EnsureSuccessStatusCode();
                homeworkViews =
response.Content.ReadAsAsync<List<HomeworkView>>().Result;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void PutHomework(HomeworkView view)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.PutAsJsonAsync($"api/PutHomework", view).Result;
```

```
                response.EnsureSuccessStatusCode();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void PostHomework(HomeworkView view)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.PostAsJsonAsync($"api/PostHomework", view).Result;
                response.EnsureSuccessStatusCode();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void DeleteHomework(int Id)
```

```
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.DeleteAsync($"api/DeleteHomework?Id={Id}").Result;
                response.EnsureSuccessStatusCode();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        #endregion

        public void GetJournal()
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetJournal").Result;
                response.EnsureSuccessStatusCode();
                journalViews =
response.Content.ReadAsAsync<List<JournalView>>().Result;
            }
            catch (Exception ex)
```

```
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void PutJournal(List<JournalView> view)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.PutAsJsonAsync($"api/PutJournal", view).Result;
                response.EnsureSuccessStatusCode();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void PutHomeworkDone(HomeworkDoneView view)
        {
            try
            {
```

```
                HttpResponseMessage response =
settingServer.client.PutAsJsonAsync($"api/PutHomeworkDone", view).Result;
                response.EnsureSuccessStatusCode();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
            }
        }

        public void GetHomeworkDone(string title)
        {
            try
            {
                HttpResponseMessage response =
settingServer.client.GetAsync($"api/GetHomeworkDone?Title={title}").Result;
                response.EnsureSuccessStatusCode();
                homeworkDoneTeacherViews =
response.Content.ReadAsAsync<List<HomeworkDoneTeacherView>>().Result;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString(), "Errors",
MessageBoxButton.OK, MessageBoxImage.Error);
                //Environment.Exit(-68071);
                //Environment.FailFast("Error", ex.InnerException);
```

```
        }
    }
}
```

SettingServer.cs

```
internal sealed class SettingServer
{
    private const string API_ADDR = "http://localhost:55570/";

    public HttpClient client;

    public SettingServer()
    {
        client = new HttpClient();
        client.BaseAddress = new Uri(API_ADDR);
        client.DefaultRequestHeaders.Accept.Clear();
        client.DefaultRequestHeaders.Accept.Add(
            new MediaTypeWithQualityHeaderValue("application/json"));
    }
}
```

HomeViewModel.cs

```
public class HomeworkViewModel
    {
        public HomeworkViewModel(HomeworkView _CodeBehind)
        {
            CodeBehind = _CodeBehind;

            CodeBehind.addlesson.Click += Addlesson_Click;
            CodeBehind.deletelesson.Click += Deletelesson_Click;
            CodeBehind.savelesson.Click += Savelesson_Click;
            CodeBehind.openbtn.Click += Openbtn_Click;
            CodeBehind.sendwork.Click += Sendwork_Click;
            CodeBehind.checkwork.Click += Checkwork_Click;
            CodeBehind.planlist.SelectionChanged +=
Planlist_SelectionChanged;

            if (CodeBehind.parent.user.role == "Teacher")
            {
                CodeBehind.addlesson.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.deletelesson.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.urlbox.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.savelesson.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.openbtn.Visibility =
System.Windows.Visibility.Collapsed;
```

```
                CodeBehind.sendwork.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.checkwork.Visibility =
System.Windows.Visibility.Visible;
            }
            else
            {
                CodeBehind.addlesson.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.deletelesson.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.urlbox.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.savelesson.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.openbtn.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.sendwork.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.checkwork.Visibility =
System.Windows.Visibility.Collapsed;
            }

            MaterialLoad();
        }


        public HomeworkView CodeBehind;
```

```
#region Load

public void MaterialLoad()
{
    CodeBehind.parent.controller.GetHomework();

    List<string> tmplist = new List<string>();

    foreach (var item in CodeBehind.parent.controller.homeworkViews)
    {
        if (CodeBehind.parent.user.role == "Teacher")
        {
            tmplist.Add(item.Title);
        }
        else
        {
            int tmp =
CodeBehind.parent.controller.GetClass(CodeBehind.parent.user.id);
            if (tmp == item.Class)
            {
                tmplist.Add(item.Title);
            }
        }

    }

    CodeBehind.planlist.ItemsSource = tmplist;
```

```
            CodeBehind.contentlesson.Visibility =
System.Windows.Visibility.Collapsed;
        }

        #endregion

        #region Save

        private void Savelesson_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            CodeBehind.parent.controller.homeworkViews.FirstOrDefault(x =>
x.Title == CodeBehind.planlist.SelectedItem.ToString()).Url =
CodeBehind.urltxt.Text;


CodeBehind.parent.controller.PutHomework(CodeBehind.parent.controller.homew
orkViews.FirstOrDefault(x => x.Title ==
CodeBehind.planlist.SelectedItem.ToString()));
        }

        private void Addlesson_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            AddHomework addHomework = new AddHomework();
            if (addHomework.ShowDialog() == true)
            {
                MaterialLoad();
            }
```

```
        }

        private void Deletelesson_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            if (CodeBehind.planlist.SelectedItem != null)
            {

CodeBehind.parent.controller.DeleteMaterial(CodeBehind.parent.controller.home
workViews.FirstOrDefault(x => x.Title ==
CodeBehind.planlist.SelectedItem.ToString()).Id);

                MaterialLoad();
            }
        }

        #endregion

        private void Planlist_SelectionChanged(object sender,
System.Windows.Controls.SelectionChangedEventArgs e)
        {
            if (CodeBehind.planlist.SelectedItem != null)
            {
                CodeBehind.contentlesson.Visibility =
System.Windows.Visibility.Visible;

                CodeBehind.title.Content =
CodeBehind.parent.controller.homeworkViews.FirstOrDefault(x => x.Title ==
CodeBehind.planlist.SelectedItem.ToString()).Title;
```

```
            }
        }

        private void Openbtn_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            if (CodeBehind.planlist.SelectedItem != null)
            {

System.Diagnostics.Process.Start(CodeBehind.parent.controller.homeworkViews.
FirstOrDefault(x => x.Title == CodeBehind.planlist.SelectedItem.ToString()).Url);
            }
        }

        private void Sendwork_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            HomeworkDone homeworkDone = new
HomeworkDone(CodeBehind.parent.user,
CodeBehind.planlist.SelectedItem.ToString());
            if (homeworkDone.ShowDialog() == true)
            {
                MaterialLoad();
            }
        }

        private void Checkwork_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
```

```
            HomeworksList homeworksList = new
HomeworksList(CodeBehind.planlist.SelectedItem.ToString());
            if (homeworksList.ShowDialog() == true)
            {
                MaterialLoad();
            }
        }
    }
```

JournalViewModel.cs

```
public class JournalViewModel
    {
        public JournalViewModel(Journal _CodeBehind)
        {
            CodeBehind = _CodeBehind;

            CodeBehind.savelesson.Click += Savelesson_Click;

            JournalLoad();
        }

        public Journal CodeBehind;

        #region Load

        public void JournalLoad()
        {
```

```
            CodeBehind.parent.controller.GetJournal();


            if (CodeBehind.parent.user.role == "Teacher")
            {
               CodeBehind.journalgrid.IsReadOnly = false;
               CodeBehind.savelesson.Visibility =
System.Windows.Visibility.Visible;
            }
            else
            {
               CodeBehind.journalgrid.IsReadOnly = true;
               CodeBehind.savelesson.Visibility =
System.Windows.Visibility.Collapsed;

               int tmp =
CodeBehind.parent.controller.GetClass(CodeBehind.parent.user.id);

               CodeBehind.parent.controller.journalViews.RemoveAll(x =>
x.Class != tmp);
            }

            CodeBehind.journalgrid.ItemsSource =
CodeBehind.parent.controller.journalViews;

         }

         #endregion
```

```
        #region Save

        private void Savelesson_Click(object sender,
System.Windows.RoutedEventArgs e)
        {

CodeBehind.parent.controller.PutJournal(CodeBehind.parent.controller.journalVie
ws);

            JournalLoad();
        }

        #endregion

    }
```

MainWindowView.cs

```
    class MainWindowView
    {
        public MainWindowView(MainWindow _CodeBehind)
        {
            CodeBehind = _CodeBehind;

            CodeBehind.btnExit.Click += BtnExit_Click;
            CodeBehind.btnMin.Click += BtnMin_Click;
            CodeBehind.sidePanel.MouseDown += Grid_MouseDown;
```

```
            CodeBehind.materials.MouseDoubleClick +=
Menu_MouseDoubleClick;
            CodeBehind.homework.MouseDoubleClick +=
Menu_MouseDoubleClick;
            CodeBehind.journal.MouseDoubleClick +=
Menu_MouseDoubleClick;

            Authorization authorization = new Authorization(CodeBehind);
            authorization.ShowDialog();

            CodeBehind.PersonName.Content = CodeBehind.user.name;
        }

        public MainWindow CodeBehind;

        #region General

        private void MoveCursorMenu(int index)
        {
            CodeBehind.TrainsitionigContentSlide.OnApplyTemplate();
            int m = (10 + (50 * index));
            CodeBehind.GridCursor.Margin = new Thickness(0, m, 0, 0);
        }

        private void BtnMin_Click(object sender, RoutedEventArgs e)
        {
            CodeBehind.WindowState = WindowState.Minimized;
        }
```

```
        private void BtnExit_Click(object sender, RoutedEventArgs e)
        {
            MessageBoxResult result = MessageBox.Show("Are you sure to
close the application?", "Exit", MessageBoxButton.YesNo,
MessageBoxImage.Information);
            if (result == MessageBoxResult.Yes) CodeBehind.Close();
        }

        private void Grid_MouseDown(object sender,
System.Windows.Input.MouseButtonEventArgs e)
        {
            if (e.LeftButton == MouseButtonState.Pressed)
            {
                CodeBehind.DragMove();
            }
        }

        #endregion

        #region Menu
        private void Menu_MouseDoubleClick(object sender,
MouseButtonEventArgs e)
        {
            ListViewItem item = (ListViewItem)sender;

            switch (item.Name.ToString())
            {
                case "materials":
                    {
```

```
                    MoveCursorMenu(0);

                    MaterialsView materialsView = new
MaterialsView(CodeBehind);
                    //отображаем
                    CodeBehind.ContentPages.Content = materialsView;

                    break;
                }
            case "homework":
                {
                    MoveCursorMenu(1);

                    HomeworkView homeworkView = new
HomeworkView(CodeBehind);
                    //отображаем
                    CodeBehind.ContentPages.Content = homeworkView;

                    break;
                }
            case "journal":
                {
                    MoveCursorMenu(2);

                    Journal journalView = new Journal(CodeBehind);
                    //отображаем
                    CodeBehind.ContentPages.Content = journalView;

                    break;
```

```
                }
            default:
                break;
        }
    }

    #endregion
}
```

MaterialsViewModel.cs

```
public class MaterialsViewModel
    {
        public MaterialsViewModel(MaterialsView _CodeBehind)
        {
            CodeBehind = _CodeBehind;

            CodeBehind.addlesson.Click += Addlesson_Click;
            CodeBehind.deletelesson.Click += Deletelesson_Click;
            CodeBehind.savelesson.Click += Savelesson_Click;
            CodeBehind.planlist.SelectionChanged +=
Planlist_SelectionChanged;

            if (CodeBehind.parent.user.role == "Teacher")
            {
                CodeBehind.addlesson.Visibility =
System.Windows.Visibility.Visible;
```

```
                CodeBehind.deletelesson.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.savelesson.Visibility =
System.Windows.Visibility.Visible;
                CodeBehind.materialcontent.IsReadOnly = false;
            }
            else
            {
                CodeBehind.addlesson.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.deletelesson.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.savelesson.Visibility =
System.Windows.Visibility.Collapsed;
                CodeBehind.materialcontent.IsReadOnly = true;
            }

            MaterialLoad();
        }


        public MaterialsView CodeBehind;

        #region Load

        public void MaterialLoad()
        {
            CodeBehind.parent.controller.GetMaterial();
```

```
            List<string> tmplist = new List<string>();

            foreach (var item in CodeBehind.parent.controller.lessonsViews)
            {
                if (CodeBehind.parent.user.role == "Teacher")
                {
                    tmplist.Add(item.Title);
                }
                else
                {
                    int tmp =
CodeBehind.parent.controller.GetClass(CodeBehind.parent.user.id);
                    if (tmp == item.Class)
                    {
                        tmplist.Add(item.Title);
                    }
                }

            }

            CodeBehind.planlist.ItemsSource = tmplist;

            CodeBehind.contentlesson.Visibility =
System.Windows.Visibility.Collapsed;
        }

        #endregion

        #region Save
```

```
        private void Savelesson_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            CodeBehind.parent.controller.lessonsViews.FirstOrDefault(x =>
x.Title == CodeBehind.planlist.SelectedItem.ToString()).Title =
CodeBehind.title.Content.ToString();
            CodeBehind.parent.controller.lessonsViews.FirstOrDefault(x =>
x.Title == CodeBehind.planlist.SelectedItem.ToString()).Content =
CodeBehind.materialcontent.Text;


CodeBehind.parent.controller.PutMaterial(CodeBehind.parent.controller.lessonsVi
ews.FirstOrDefault(x => x.Title ==
CodeBehind.planlist.SelectedItem.ToString()));
        }

        private void Addlesson_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            AddLesson addLesson = new AddLesson();
            if (addLesson.ShowDialog() == true)
            {
                MaterialLoad();
            }
        }
```

```
        private void Deletelesson_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            if(CodeBehind.planlist.SelectedItem != null)
            {

CodeBehind.parent.controller.DeleteMaterial(CodeBehind.parent.controller.lesson
sViews.FirstOrDefault(x => x.Title ==
CodeBehind.planlist.SelectedItem.ToString()).Id);

                MaterialLoad();
            }
        }

        #endregion

        private void Planlist_SelectionChanged(object sender,
System.Windows.Controls.SelectionChangedEventArgs e)
        {
            if (CodeBehind.planlist.SelectedItem != null)
            {
                CodeBehind.contentlesson.Visibility =
System.Windows.Visibility.Visible;

                CodeBehind.title.Content =
CodeBehind.parent.controller.lessonsViews.FirstOrDefault(x => x.Title ==
CodeBehind.planlist.SelectedItem.ToString()).Title;
```

```
                CodeBehind.materialcontent.Text =
CodeBehind.parent.controller.lessonsViews.FirstOrDefault(x => x.Title ==
CodeBehind.planlist.SelectedItem.ToString()).Content;
            }
        }
    }
}
```

AuthorizationLogic.cs

```
public class AuthorizationLogic
    {
        private SchoolDBEntities db;
        public AuthorizationLogic()
        {
            db = new SchoolDBEntities();
        }

        internal User GetAuthorization(string userLogin, string userPassword)
        {
            User userView = new User();
            User tmpUser = db.Users.FirstOrDefault(u => u.login == userLogin
&& u.password == userPassword);
            if (tmpUser != null)
            {
                userView.id = tmpUser.id;
                userView.name = tmpUser.name;
                userView.surname = tmpUser.surname;
                userView.login = tmpUser.login;
```

```
            userView.password = tmpUser.password;
            userView.email = tmpUser.email;
            userView.birthday = tmpUser.birthday;

            userView.role = tmpUser.role;
        }
        return userView;
    }

    internal string GetRestoreAccess(string email)
    {
        string AnswerReq = "Email was not found";
        if (!String.IsNullOrWhiteSpace(email))
        {
            User tmpUser = db.Users.FirstOrDefault(x =>
x.email.ToLower().TrimStart().TrimEnd() ==
email.ToLower().TrimStart().TrimEnd());
            if (tmpUser != null)
            {
                SendEmail(tmpUser.email, tmpUser.login, tmpUser.password);
                AnswerReq = "Username and password have been sent to your
email. Good luck!";
            }
        }
        return AnswerReq;
    }

    public void SendEmail(string email, string login, string password)
    {
```

```
                var mail = "mega.finopis@ukr.net";
                var host = "smtp.ukr.net";
                var user = "mega.finopis@ukr.net";
                var pass = "sDXMR8cQpQ2CXR_";

                //Generate Message
                var mymessage = new MimeMailMessage();
                mymessage.From = new MimeMailAddress(mail);
                mymessage.To.Add(email);

                mymessage.Subject = "Restore access";
                mymessage.Body = "Добрый день!\nВам логин или пароль \n" +
"LOGIN: " + login + "\n" + "PASSWORD: " + password + "\nGood luck!\n";

                //Create Smtp Client
                var mailer = new MimeMailer(host, 465);
                mailer.User = user;
                mailer.Password = pass;
                mailer.SslType = SslMode.Ssl;
                mailer.AuthenticationMode = AuthenticationType.Base64;

                mailer.SendMailAsync(mymessage);


            }

        }
```

MainLogic.cs

```
public class MainLogic
   {
      private SchoolDBEntities db;

      public MainLogic()
      {
         db = new SchoolDBEntities();
      }

      internal int GetClass(int Id)
      {
         return db.Students.FirstOrDefault(y => y.user_id ==
Id).Class.number;
      }

      #region Material

      internal List<LessonsView> GetMaterials()
      {
         List<LessonsView> lessonsView = new List<LessonsView>();

         foreach (var item in db.Lessons)
         {
            lessonsView.Add(new LessonsView()
            {
               Id = item.id,
```

```
                Title = item.title,
                Content = item.material,
                Subject = item.Subject.title,
                Class = item.class_number
            });
        }

        return lessonsView;
    }

    internal void PutMaterial(LessonsView view)
    {
        Lesson tmp = db.Lessons.FirstOrDefault(x => x.id == view.Id);

        tmp.title = view.Title;
        tmp.material = view.Content;

        db.SaveChanges();
    }

    internal void AddMaterial(LessonsView view)
    {
        Lesson lesson = new Lesson();

        lesson.title = view.Title;
        lesson.material = view.Content;
        lesson.subject_id = db.Subjects.FirstOrDefault(x => x.title ==
view.Subject).id;
        lesson.class_number = view.Class;
```

```
        db.Lessons.Add(lesson);

        db.SaveChanges();
    }

    internal void DeleteMaterial(int Id)
    {
        db.Lessons.Remove(db.Lessons.FirstOrDefault(x => x.id == Id));

        db.SaveChanges();
    }

    internal List<string> GetSubjects()
    {
        List<string> subjects = new List<string>();

        foreach (var item in db.Subjects)
        {
            subjects.Add(item.title);
        }

        return subjects;
    }

    #endregion

    #region Homework
```

```
        internal List<HomeworkView> GetHomework()
        {
            List<HomeworkView> homeworkViews = new
List<HomeworkView>();

            foreach (var item in db.Homework)
            {
                homeworkViews.Add(new HomeworkView()
                {
                    Id = item.id,
                    Title = item.title,
                    Url = item.value,
                    Subject = item.Subject.title,
                    Class = item.class_number
                });
            }

            return homeworkViews;
        }

        internal void PutHomework(HomeworkView view)
        {
            Homework tmp = db.Homework.FirstOrDefault(x => x.id ==
view.Id);

            tmp.title = view.Title;
            tmp.value = view.Url;

            db.SaveChanges();
```

```
        }

        internal void AddHomework(HomeworkView view)
        {
            Homework homework = new Homework();

            homework.title = view.Title;
            homework.value = view.Url;
            homework.subject_id = db.Subjects.FirstOrDefault(x => x.title ==
view.Subject).id;
            homework.class_number = view.Class;

            db.Homework.Add(homework);

            db.SaveChanges();
        }

        internal void DeleteHomework(int Id)
        {
            db.Homework.Remove(db.Homework.FirstOrDefault(x => x.id ==
Id));

            db.SaveChanges();
        }

        internal void PutHomeworkDone(HomeworkDoneView view)
        {
            db.Homework_Done.Add(new Homework_Done()
            {
```

```
                homework_id = db.Homework.FirstOrDefault(x=>x.title ==
view.HomeworkTitle).id,
                student_id = db.Students.FirstOrDefault(x=>x.user_id ==
view.UserID).id,
                value = view.Url
            });

            db.SaveChanges();
        }

        internal List<HomeworkDoneTeacherView>
GetHomeworkDone(string title)
        {
            List<HomeworkDoneTeacherView> view = new
List<HomeworkDoneTeacherView>();

            foreach (var item in
db.Homework_Done.Where(x=>x.Homework.title == title))
            {
                view.Add(new HomeworkDoneTeacherView()
                {
                    Name = item.Student.User.name,
                    Url = item.value
                });
            }

            return view;
        }
```

```
#endregion

#region Journal

internal List<JournalView> GetJournal()
{
   List<JournalView> journalViews = new List<JournalView>();

   foreach (var item in db.Students)
   {
      JournalView tmp = new JournalView();

      tmp.Name = item.User.name;
      tmp.Surname = item.User.surname;
      tmp.Class = item.Class.number;

      foreach (var it in db.Journals.Where(x=>x.student_id == item.id))
      {
         if (it.Subject.title == "Математика")
         {
            tmp.Mat = it.assessment;
         }
         else if (it.Subject.title == "Физика")
         {
            tmp.Fith = it.assessment;
         }
         else if (it.Subject.title == "Украинский язык")
         {
            tmp.UkrLan = it.assessment;
```

```
            }
            else if (it.Subject.title == "Украинская литература")
            {
                tmp.UkrLit = it.assessment;
            }
            else if (it.Subject.title == "Английский язык")
            {
                tmp.Eng = it.assessment;
            }
            else if (it.Subject.title == "Русский язык")
            {
                tmp.Rus = it.assessment;
            }
            else if (it.Subject.title == "Зарубежная литература")
            {
                tmp.Lit = it.assessment;
            }
            else if (it.Subject.title == "Химия")
            {
                tmp.Him = it.assessment;
            }
            else if (it.Subject.title == "Биология")
            {
                tmp.Bio = it.assessment;
            }
        }

        journalViews.Add(tmp);
    }
```

```
            return journalViews;
        }

        internal void PutJournal(List<JournalView> journalViews)
        {
            foreach (var item in journalViews)
            {
                if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Математика") ==
null)
                {
                    db.Journals.Add(new Journal()
                    {
                        student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                        subject_id = db.Subjects.FirstOrDefault(x=>x.title ==
"Математика").id,
                        assessment = item.Mat
                    });
                }
                else
                {
                    db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title ==
"Математика").assessment = item.Mat;
```

```
                }

                if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Физика") == null)
                {
                    db.Journals.Add(new Journal()
                    {
                        student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                        subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Физика").id,
                        assessment = item.Fith
                    });
                }
                else
                {
                    db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title ==
"Физика").assessment = item.Fith;

                }

                if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
```

```
x.Student.Class.number == item.Class && x.Subject.title == "Украинский язык")
== null)
                {
                    db.Journals.Add(new Journal()
                    {
                        student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                        subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Украинский язык").id,
                        assessment = item.UkrLan
                    });
                }
                else
                {
                    db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Украинский
язык").assessment = item.UkrLan;

                }

                if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Украинская
литература") == null)
                {
                    db.Journals.Add(new Journal()
                    {
```

```
                    student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                    subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Украинская литература").id,
                    assessment = item.UkrLit
                });
            }
            else
            {
                db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Украинская
литература").assessment = item.UkrLit;

            }

            if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Английский язык")
== null)
            {
                db.Journals.Add(new Journal()
                {
                    student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                    subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Английский язык").id,
```

```
                    assessment = item.Eng
                });
            }
            else
            {
                db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Английский
язык").assessment = item.Eng;

            }

            if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Русский язык") ==
null)
            {
                db.Journals.Add(new Journal()
                {
                    student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                    subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Русский язык").id,
                    assessment = item.Rus
                });
            }
            else
            {
```

```
                db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Русский
язык").assessment = item.Rus;


            }


            if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Зарубежная
литература") == null)
            {
                db.Journals.Add(new Journal()
                {
                    student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                    subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Зарубежная литература").id,
                    assessment = item.Lit
                });
            }
            else
            {
                db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Зарубежная
литература").assessment = item.Lit;
```

```
                }

                if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Химия") == null)
                {
                    db.Journals.Add(new Journal()
                    {
                        student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                        subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Химия").id,
                        assessment = item.Him
                    });
                }
                else
                {
                    db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Химия").assessment
= item.Him;

                }

                if (db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title == "Биология") == null)
                {
```

```
                    db.Journals.Add(new Journal()
                    {
                        student_id = db.Students.FirstOrDefault(x => x.User.name
== item.Name && x.User.surname == item.Surname && x.Class.number ==
item.Class).id,
                        subject_id = db.Subjects.FirstOrDefault(x => x.title ==
"Биология").id,
                        assessment = item.Bio
                    });
                }
                else
                {
                    db.Journals.FirstOrDefault(x => x.Student.User.name ==
item.Name && x.Student.User.surname == item.Surname &&
x.Student.Class.number == item.Class && x.Subject.title ==
"Биология").assessment = item.Bio;

                }
            }

            db.SaveChanges();
        }

        #endregion
    }
}
```

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Кафедра автоматизованих систем обробки інформації та управління

УЗГОДЖЕНО
Керівник проекту
_____ *П. Ю.Радіонов*
(підпис) (ініціали, прізвище)
“13” квітня 2020 р.

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *О.А. Павлов*
(підпис) (ініціали, прізвище)
“14” квітня 2020 р.

ІНФОРМАЦІЙНА СИСТЕМА ПІДТРИМКИ ДИСТАНЦІЙНОЇ ОСВІТИ

ТЕХНІЧНЕ ЗАВДАННЯ

Шифр *6121.01.000 ТЗ*

на 8 сторінках

Київ – 2020 року

ЗМІСТ

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Назва розробки : Платформа для контролю дистанційної освіти

1.2 Організація-замовник : НТУУ «КПІ» імені І. Сікорського, ФІОТ, кафедра АСОІУ

1.3 Згідно з ГОСТ 34.602-89 ТЗ є основним документом, що визначає вимоги і порядок створення (розвитку або модернізації) інформаційної системи, відповідно до якого проводиться її розробка і приймання при введенні в дію.

1.4 Початок робіт : 15.03.2020р. Закінчення робіт : 25.05.2020р.

2 ПРИЗНАЧЕННЯ І ЦІЛІ СТВОРЕННЯ СИСТЕМИ

2.1 Призначення цієї розробки є автоматизації навчального процесу, зменшення навантаження на викладачів, прискорення роботи навчального закладу та якості надання матеріалу.

2.2 Метою розробки є покращення навчального процесу у закладах середньої освіти.

Задачі :

- створення бази даних;
- розробка WebApi сервісу;
- розробка Desktop застосунку;
- розробка мобільного застосунку
- розробка веб застосунку
- з’єднання застосунків з сервісом;
- автоматизація оновлення та синхронізації.

3 ХАРАКТЕРИСТИКА ОБ'ЄКТА АВТОМАТИЗАЦІЇ

Предметною областю є робота закладу середньої або початкової освіти, зокрема автоматизація навчального процесу.

Планується автоматизація таких процесів як:

- приймання домашнього завдання;
- проведення самостійних та контрольних робіт;
- підрахунок поточних та підсумкових балів учнів;
- ведення статистики успішності та поведінки учнів;
- ведення (не планування) розкладу та можливість його зміни;
- надання матеріалу з уроків що біли проведені або ще будуть;
- надання матеріалу за конкретною темою;
- проведення батьківських зборів та контроль успішності дітей.

4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Вимоги до функціональних характеристик

4.1.1 Програмне забезпечення повинно забезпечувати виконання наступних основних функції:

4.1.1.1 Для користувача:

− можливість централізованого контролю над проходженням іспиту для викладача;

− можливість перегляду історії складених іспитів;

− можливість вибору мови програмування для реалізації екзаменаційного завдання для студента;

− наявність зручного інструменту для написання, запуску та перегляду програмного коду.

4.1.2 Розробку виконати на платформі Linux

4.2 Вимоги до надійності

4.2.1 Передбачити контроль введення інформації.

4.2.2 Передбачити захист від некоректних дій користувача.

4.2.3 Забезпечити цілісність інформації в базі даних.

4.3 Умови експлуатації

4.3.1 Умови експлуатації згідно СанПін 2.2.2.542 – 96.

4.3.2 Обслуговування

Даний продукт не потребує регулярного обслуговування.

4.3.3 Обслуговуючий персонал

Даний продукт не потребує залучення обслуговуючого персоналу.

4.4 Вимоги до складу і параметрів технічних засобів

4.4.1 Програмне забезпечення повинно функціонувати на IBM-сумісних персональних комп‘ютерах.

4.4.2 Мінімальна конфігурація технічних засобів:

Тип процесору x86_64 або ARM v8 64.

Об‘єм ОЗП 1024 Мб.

1 Гб постійного дискового простору

Клавіатура

ЖК-дісплей 4.5

Вимоги до інформаційної та програмної сумісності

4.5.1 Програмне забезпечення повинно працювати під управлінням операційних систем сімейств Unix та Windows.

4.5.2 Вхідні дані є набором текстових документів, що являють собою сирцевий програмний код.

4.5.3 Результати представлені у форматі JSON-подібних документів колекцій MongoDB.

4.5.4 Програмне забезпечення повинно надавати графічний інтерфейс користувача та REST API в будь-якому текстовому чи бінарному форматі.

4.6 Вимоги до маркування та пакування

Вимоги до маркування та пакування не пред‘являються.

4.7 Вимоги до транспортування та зберігання

Вимоги до транспортування та зберігання не пред'являються.

4.8 Спеціальні вимоги

Спеціальні вимоги не пред’являються.

5 СТАДІЇ І ЕТАПИ РОЗРОБКИ

№	Назва етапу	Строк	Звітність
1	Розробка технічного завдання	25.03.2020	Технічне завдання
2	Аналіз вимог та уточнення специфікацій	01.04.2020	Специфікації програмного забезпечення
3	Проектування структури програмного забезпечення, проектування компонентів	15.04.2020	Схема структурна варіантів використання, схема структурна баз даних, схема структурна потоків даних
4	Програмна реалізація програмного забезпечення	30.04.2020	Тексти програмного забезпечення
5	Тестування програмного забезпечення	05.05.2020	Тести, результати тестування
6	Розробка матеріалів текстової частини проекту	17.05.2020	Пояснювальна записка
7	Розробка матеріалів графічної частини проекту	23.05.2020	Графічний матеріал проекту
8	Оформлення технічної документації проекту	25.05.2020	Технічна документація

6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ

6.1 Види випробувань

Тестування готових програмних засобів має бути виконано згідно документа «Програми та методики тестування».

Графічний матеріал
до дипломного проєкту

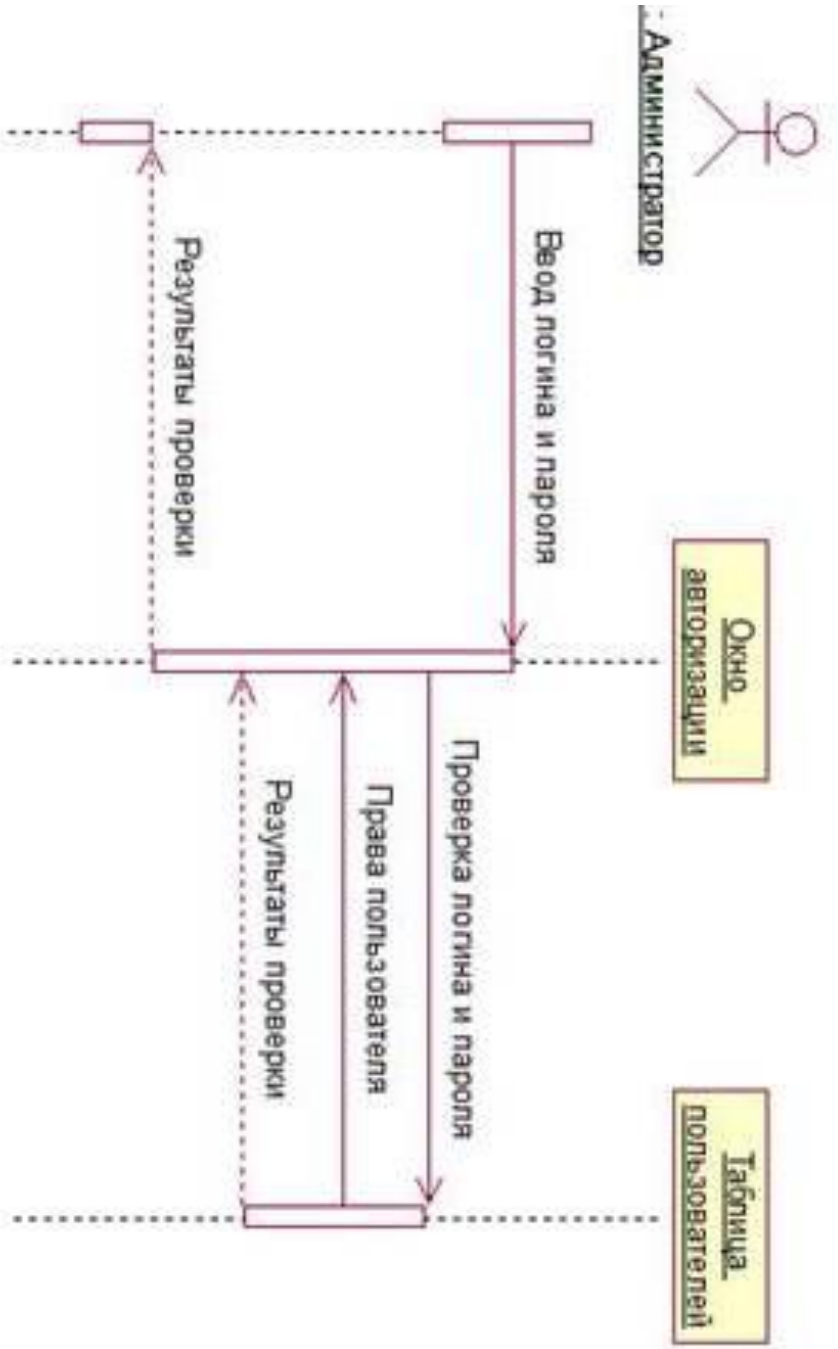
на тему: Інформаційна система підтримки дистанційної освіти

Київ – 2020 року

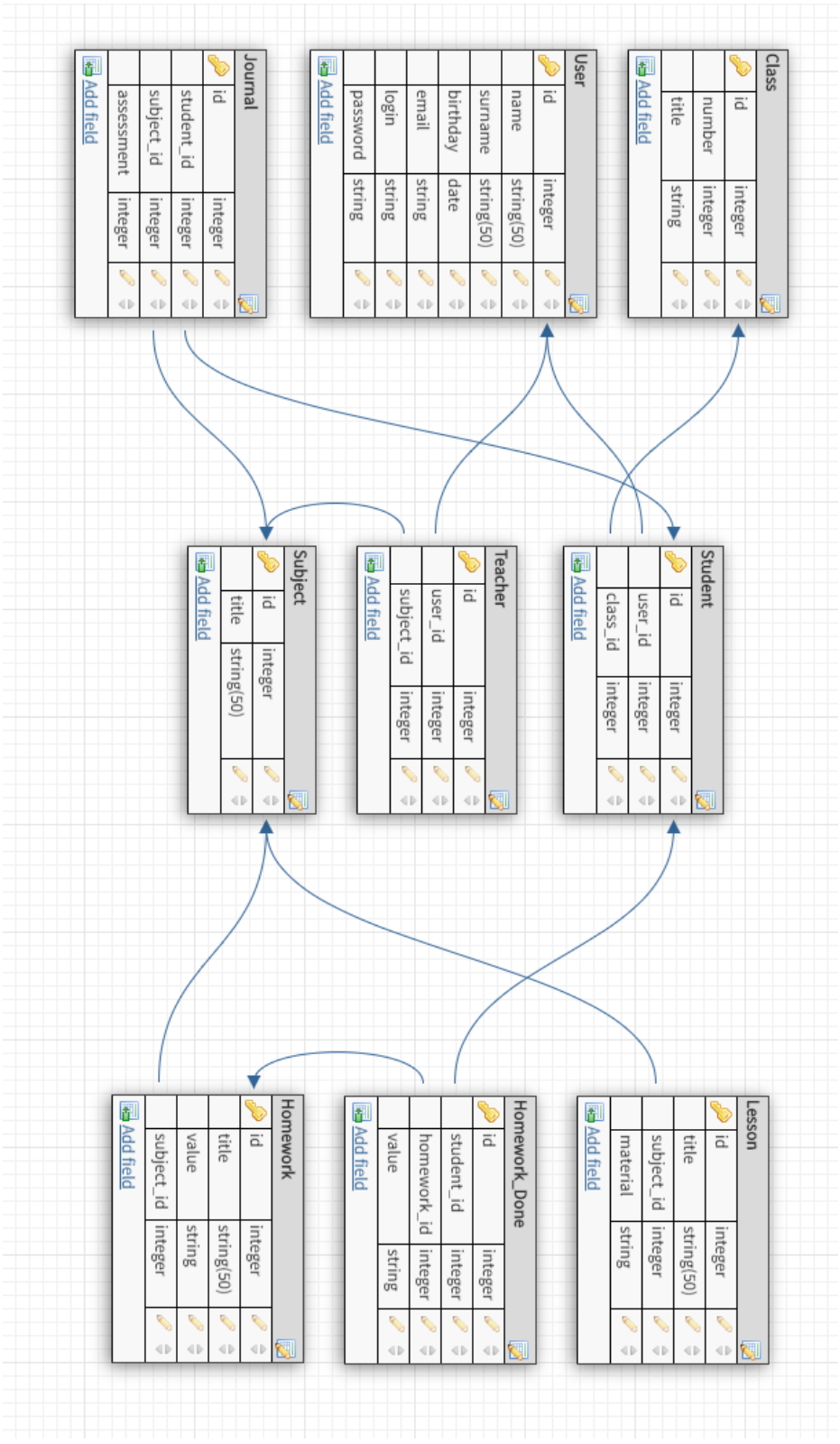
					ДП 6121.05.000 ССК	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна класів	Літера / Маса / Масштаб
Розроб.		Сіваченко В.О.				
Перевірив.		Радіонов П.Ю.				Аркуш 1 / Аркушів 2
Н. кон.		Тєлишева Т.О.			Інформаційна система підтримки дистанційної освіти	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-з61
Затв.		Радіонов П.Ю.				

ДП 6121.05.000 ССК

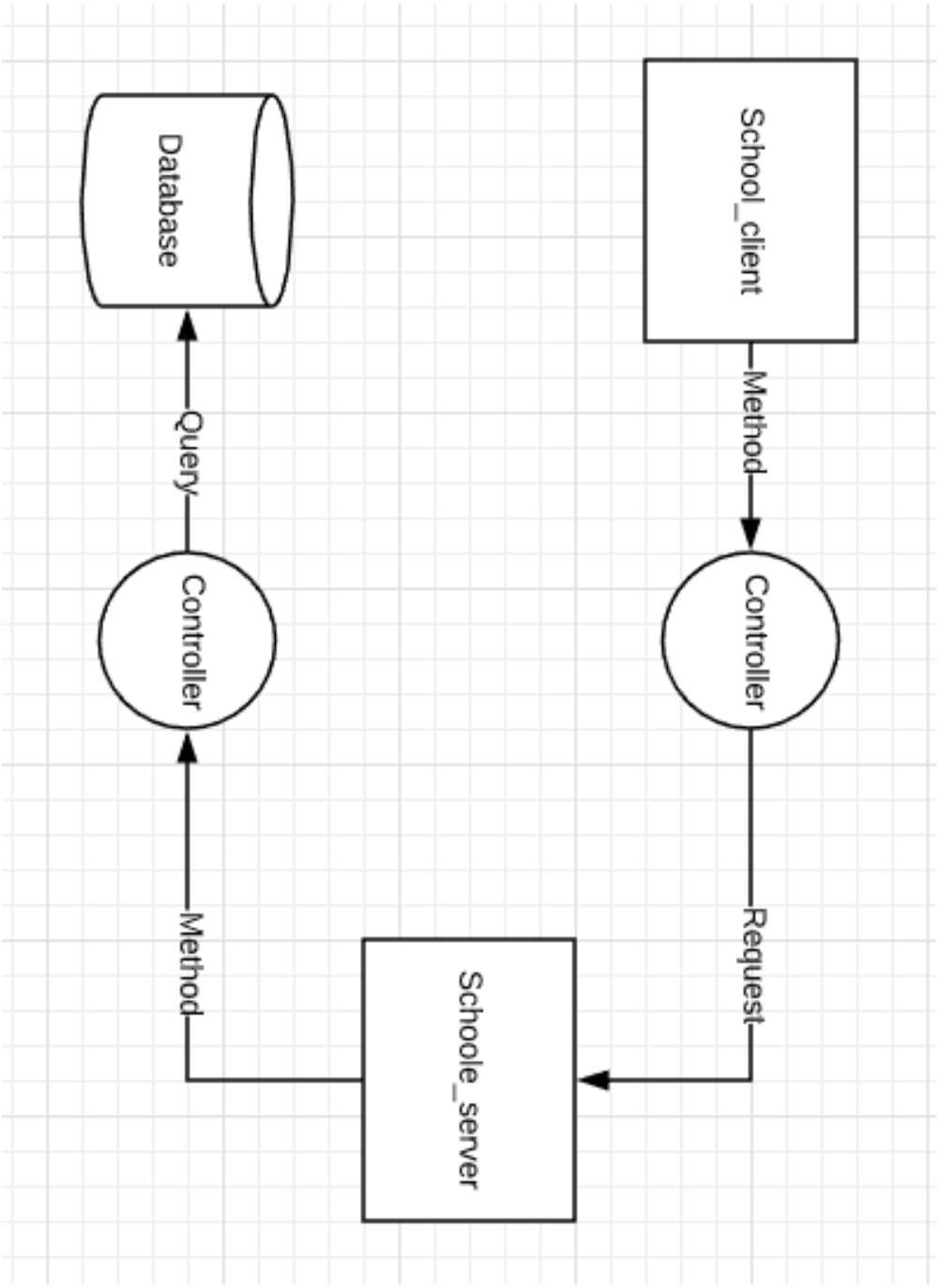
					ДП 6121.05.000 ССК	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна класів	Літера / Маса / Масштаб
Розроб.		Сіваченко В.О.				
Перевірив.		Радіонов П.Ю.				Аркуш 2 / Аркушів 2
Н. кон.		Тєлишева Т.О.			Інформаційна система підтримки дистанційної освіти	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-з61
Затв.		Радіонов П.Ю.				



					ДП 6121.06.000 ССП		
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна послідовності	Літера	Маса Масштаб
Розроб.		Сіваченко В.О.					
Перевірив.		Радіонов П.Ю.					
						Аркуш 1	Аркушів 1
Н. кон.		Тєлишева Т.О.			Інформаційна система підтримки дистанційної освіти	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-з61	
Затв.		Радіонов П.Ю.					



					ДП 6121.04.000 СБД	
Зм.	Арк.	№ документа	Підпис	Дата	Схема бази даних	Літера / Маса / Масштаб
Розроб.		Сіваченко В.О.				
Перевірив.		Радіонов П.Ю.				Аркуш 1 / Аркушів 1
Н. кон.		Тєлишева Т.О.			Інформаційна система підтримки дистанційної освіти	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-з61
Затв.		Радіонов П.Ю.				



					ДП 6121.07.000 СК		
						Літера	Маса Масштаб
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна компонентів програмного забезпечення		
Розроб.		Сіваченко В.О.				Аркуш 1	Аркушів 1
Перевірив.		Радіонов П.Ю.					
Н. кон.		Тєлишева Т.О.			Інформаційна система підтримки дистанційної освіти	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-з61	
Затв.		Радіонов П.Ю.					